Exploiting Concatenation in the Design of Low-Density Parity-Check Codes

Marco Baldi, Giovanni Cancellieri, and Franco Chiaraluce
DIBET, Polytechnic University of Marche,
Ancona, Italy
Email: {m.baldi, g.cancellieri, f.chiaraluce}@univpm.it

***Abstract*—This paper describes a method to exploit the concatenation of very simple component codes in order to obtain good low-density parity-check codes. This allows to design codes having a number of benefits, as high flexibility in length and rate and low encoding complexity. We focus on two kinds of concatenation: the former is classic serial concatenation, in which redundancy is progressively added to the encoded word, whereas the latter is the special case of concatenation coinciding with a bi-dimensional product code. The proposed design technique allows to obtain codes characterized by parity-check matrices with a low density of 1 symbols and free of short cycles in their associated Tanner graph; so efficient algorithms based on the belief propagation principle can be adopted for their decoding. In addition, the systematic form of the component codes can ensure rate compatibility; so the proposed codes can be adopted in type-II hybrid automatic repeat request schemes. We analyze their properties through theoretical arguments and provide some design examples to assess their performance.**

***Keywords*-LDPC codes; concatenated codes; product codes.**

I. INTRODUCTION

This paper deals with the design of a family of structured low-density parity-check (LDPC) codes based on serial concatenation [1]. The introduction of concatenation in the design of efficient schemes for forward error correction (FEC) is due to Dave Forney in 1966 [2]. Since then, several forms of concatenation have been exploited in the design of codes, providing better and better performance, until the introduction of turbo codes [3], based on concatenated convolutional codes and Bahl, Cocke, Jelinek and Raviv (BCJR) decoders [4].

In recent years, LDPC codes [5] have become the state of the art in FEC techniques, due to their capacity-approaching performance under belief propagation decoding [6]. Since their recent rediscovery [7], many design techniques for LDPC codes have been proposed, that can be classified as structured and non-structured approaches. Structured approaches permit to design codes characterized by rather low implementation complexity that, however, must obey a number of constraints in terms of length and rate, as in the case of quasi-cyclic (QC) LDPC codes [8]. Non-structured designs, instead, are able to produce good LDPC codes with very fine length and rate granularity, as occurs by adopting the progressive edge growth technique [9]. However, non-structured techniques generally produce codes that are less prone to hardware implementation, due to the lack of structure in their characteristic matrices.

As a first aim of this paper, we study how to design structured LDPC codes that can be represented as the serial concatenation of very simple components [10], [11]. We adopt, as component codes, a class of polynomial codes having a binomial generator polynomial. This approach permits us to design codes characterized by a very simple intrinsic structure, that allows to adopt low complexity encoder circuits. The proposed codes can also be shortened arbitrarily, thus obtaining fine length and rate granularity.

We show that the proposed technique can be used to design sets of rate compatible codes [12]. Rate compatibility is important, for example, in the implementation of Type-II Hybrid Automatic Repeat-reQuest (T-II HARQ) schemes, where packets are initially encoded with a high rate code, and then redundancy is transmitted incrementally until successful decoding is achieved. T-II HARQ schemes are particularly useful in packet switched communication networks, since they allow the achievement of capacity-approaching unequal error correction. Serial concatenation is a consolidated procedure to design rate compatible codes [13], [14].

When the code length increases, the advantages of adopting serial concatenation can be limited by the need of rather large component codes. For this reason, we consider a further form of concatenation, coinciding with the structure of a bi-dimensional product code. At the cost of some loss in performance, the adoption of a product structure allows to keep small the size of the component codes while designing concatenated codes with large blocks. We will show that both the serial concatenation and the product structure are able to ensure the LDPC nature of the codes when adopting the special class of component codes we consider. We will also develop some examples aimed at assessing the effect of different design choices in the combination of serial concatenation with the product structure.

The paper is organized as follows. Section II introduces the code design approach based on the serial concatenation of codes with binomial generator polynomial. In Section III the characteristics of the proposed codes are studied. Section IV reports some design examples of serially concatenated codes and their simulated performance, whereas Section V gives some examples of usage of such codes

in bi-dimensional product structures. Finally, Section VI concludes the paper.

II. Code Design

A. Component codes

In the considered scheme, the i-th component code ($i = 1, \ldots, M$) is a polynomial code with generator polynomial

$$g^{(i)}(x) = (1 + x^{r_i}), \quad (1)$$

where r_i, that is a suitably chosen positive integer, represents the code redundancy. We denote by n_i the length of the i-th component code and by $k_i = n_i - r_i$ its dimension. Each component code can be seen as a shortened version of a binary cyclic code with length $N_i = \left\lceil \frac{n_i}{r_i} \right\rceil \cdot r_i \geq n_i$, where function $\lceil \cdot \rceil$ returns the smallest integer greater than or equal to its argument. It can be easily verified that:

$$\left(1 + x^{N_i}\right) = (1 + x^{r_i}) \left(1 + x^{r_i} + x^{2r_i} + \ldots + x^{N_i - r_i}\right); \quad (2)$$

so a valid parity polynomial for the cyclic code is:

$$h^{(i)}(x) = \left(1 + x^{r_i} + x^{2r_i} + \ldots + x^{N_i - r_i}\right). \quad (3)$$

Starting from the coefficients of the parity polynomial, it is easy to obtain a valid parity-check matrix for any binary cyclic code in its standard form [15]. For the considered cyclic codes, the parity-check matrix ($\mathbf{H}_i$) has a very regular structure, that is a single row of $\left\lceil \frac{n_i}{r_i} \right\rceil$ identity blocks with size $r_i \times r_i$. It follows that $\mathbf{H}_i$ has size $r_i \times N_i$.

The i-th cyclic code has dimension $K_i = N_i - r_i \geq n_i - r_i = k_i$. Each K_i-bit information vector can be associated to an information polynomial $m^{(i)}(x)$ as follows:

$$m^{(i)}(x) = m_0 + \ldots + m_{k_i - 1} x^{k_i - 1} + \ldots + m_{K_i - 1} x^{K_i - 1}, \quad (4)$$

where $m_0 \ldots m_{K_i - 1} \in \{0, 1\}$ are the information bits. The codeword corresponding to $m^{(i)}(x)$ can be expressed, in polynomial terms, as follows:

$$\begin{aligned} t^{(i)}(x) &= t_0 + \ldots + t_{n_i - 1} x^{n_i - 1} + \ldots + t_{N_i - 1} x^{N_i - 1} \\ &= m^{(i)}(x) g^{(i)}(x). \end{aligned} \quad (5)$$

We shorten the cyclic code by imposing $m_{k_i} = m_{k_i + 1} = \ldots = m_{K_i - 1} = 0$. This implies $t_{n_i} = t_{n_i + 1} = \ldots = t_{N_i - 1} = 0$, and the parity-check matrix can be accordingly shortened by eliminating its first $N_i - n_i$ columns. Figure 1 shows the structure of the parity-check matrix of the cyclic code and its shortened version. Black diagonals represent 1 symbols, whereas the other symbols are null. The shortened matrix corresponds to the sub-matrix marked in grey.

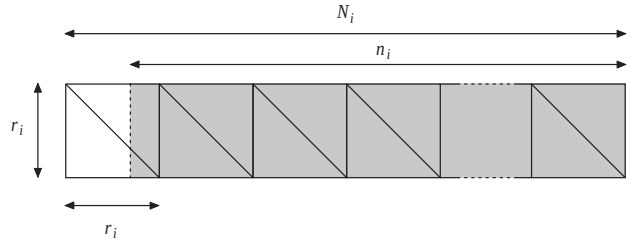

Figure 1. Parity-check matrix of the i-th component code.

B. Serial concatenation

We consider a first family of codes that are obtained as the serial concatenation of M component codes of the type described in Section II-A. We call this family of codes Multiple Serially Concatenated Multiple Parity-Check (M-SC-MPC) codes [16]. Each component code is in systematic form; so, we obtain a systematic serial concatenation, in which redundancy is incrementally appended at the end of the information vector.

The serially concatenated code has dimension k and length n. If we set $n_0 = k$, the i-th component code has dimension $k_i = n_{i-1}$, redundancy r_i and length $n_i = k_i + r_i$, with $i = 1 \ldots M$. The following relations hold:

$$\begin{aligned} n_1 &= n_0 + r_1 = k + r_1 \\ n_2 &= n_1 + r_2 = k + r_1 + r_2 \\ &\vdots \\ n_M &= n_{M-1} + r_M = k + \textstyle\sum_{i=1}^{M} r_i \end{aligned} \quad (6)$$

and the overall code has length $n = n_M$ and redundancy $r = \sum_{i=1}^{M} r_i$. The parity-check matrix of each component code, in the form of Figure 1, can be used to obtain a valid parity-check matrix for the serially concatenated code. Such matrix ($\mathbf{H}$) is in lower triangular form, and it is shown in Figure 2, for the case $M = 3$. Each column of $\mathbf{H}$ has maximum density $M/r = M/\sum_{i=1}^{M} r_i$ (that is the density of its leftmost n_1 columns); the values r_i, $i = 1 \ldots M$, must be chosen high enough as to make $\mathbf{H}$ sparse, thus obtaining an LDPC code. Furthermore, we will see in the following that, under suitable conditions, matrix $\mathbf{H}$ corresponds to a Tanner graph free of short cycles, that allows to adopt efficient LDPC decoding algorithms.

It should be noted that the proposed scheme can be seen as a generalization of the multiple serially concatenated single parity-check (M-SC-SPC) approach [17]. The latter, however, assuming $r_1 = r_2 = \ldots = r_M = 1$, does not permit to obtain LDPC codes. Moreover, the performance of M-SC-MPC codes can be better than that of M-SC-SPC codes [16].

A first requirement, when designing serially concatenated codes with the proposed technique, consists in making their parity-check matrix suitable for application of decoding

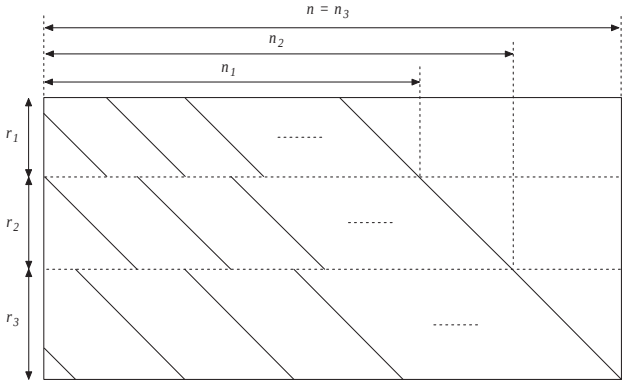

Figure 2. Parity-check matrix of the serially concatenated code.

algorithms based on the Belief Propagation (BP) principle. This can be achieved only if the associated Tanner graph is free of short cycles. For the considered codes, such condition can be easily ensured by following the rules established in Lemma 1 and Corollary 1, that are reported next.

Lemma 1: In order to obtain a Tanner graph representation free of length-4 cycles, the M-SC-MPC code must have length $n \leq n_{\max}$, with:

$$n_{\max} = \min_{\substack{i,j \in [1,M] \\ i<j}} \left\{ \mathrm{lcm}(r_i, r_j) + \sum_{l=i+1}^{M} r_l \right\}. \qquad (7)$$

Proof: Let us focus on the parity-check matrix for $M = 3$ (see Figure 2), and consider the first two blocks of rows (i.e. the first $r_1 + r_2$ rows). A length-4 cycle exists between any two rows if they have two 1 symbols at the same columns. It can be easily verified that this cannot occur when $n_1 \leq \mathrm{lcm}(r_1, r_2)$, that is $n \leq \mathrm{lcm}(r_1, r_2) + r_2 + r_3$. If we consider the first and third blocks of rows, length-4 cycles are avoided among their rows when $n_1 \leq \mathrm{lcm}(r_1, r_3)$, that is $n \leq \mathrm{lcm}(r_1, r_3) + r_2 + r_3$. Finally, in the last two blocks of rows (i.e. the last $r_2 + r_3$ rows), length-4 cycles are absent for $n_2 \leq \mathrm{lcm}(r_2, r_3)$, that is $n \leq \mathrm{lcm}(r_2, r_3) + r_3$. Hence, in order to avoid length-4 cycles in the matrix of Figure 2, it must be $n \leq \min[\mathrm{lcm}(r_1, r_2) + r_2 + r_3, \mathrm{lcm}(r_1, r_3) + r_2 + r_3, \mathrm{lcm}(r_2, r_3) + r_3]$. This confirms the validity of (7) for the case $M = 3$. The same reasoning can be easily extended to the general case of M component codes, thus proving the assertion. ■

Corollary 1: For a set of distinct, coprime and increasingly ordered r_i's, $i = 1 \ldots M$, the Tanner graph of the M-SC-MPC code is free of length-4 cycles for code length $n \leq n'_{\max}$, with:

$$n'_{\max} = r_1 r_2 + \sum_{j=2}^{M} r_j. \qquad (8)$$

Proof: If we refer again to the case $M = 3$ (see Figure 2), by Lemma 1, length-4 cycles are avoided when $n \leq \mathrm{lcm}(r_1, r_2) + r_2 + r_3 = r_1 r_2 + r_2 + r_3 = r_2(r_1 + 1) + r_3$, $n \leq \mathrm{lcm}(r_1, r_3) + r_2 + r_3 = r_1 r_3 + r_2 + r_3$ and $n \leq \mathrm{lcm}(r_2, r_3) + r_3 = r_2 r_3 + r_3$. But $r_1 r_2 < r_1 r_3$ and $r_1 + 1 < r_3$, so the first condition is the most stringent one. This result can be extended to a generic value of M, in the sense that the condition set by the first two blocks of rows is always the most stringent one. So, under the hypotheses of the corollary, Eq. (8) results. ■

It is important to observe that the proposed design technique achieves very fine granularity in the code length. In fact, provided that $n \leq n_{\max}$, each value of n is feasible and able to ensure a Tanner graph representation free of length-4 cycles.

By comparing (7) and (8), we can observe that the choice of r_i's all distinct and coprime yields the highest values for the code length, i.e. the highest flexibility in the choice of n. For this reason, in the following we will consider distinct, coprime and increasingly ordered r_i's, in such a way as to apply Eq. (8).

C. Design of product codes

A particular form of serial concatenation can be realized by constructing a product code, that can be seen as an N-dimensional polytope in which each component code works along one dimension.

We focus on the simplest form of product codes, that are bi-dimensional codes. In this case, the overall code results from two component codes working on the two dimensions of a rectangular matrix. An example of such matrix is reported in Figure 3; we denote by (n_a, k_a, r_a) and (n_b, k_b, r_b) the length, dimension and redundancy of the two component codes. The information bits are written in the inner $k_b \times k_a$ matrix by following a fixed order (for example, in row-wise manner from top left to bottom right). When the inner matrix is filled, the first component code acts on its rows, producing a set of $k_b r_a$ checks, that fill the light grey rectangular region marked as "Checks a". Then, the second component code acts on all n_a columns, so producing $k_a r_b$ checks on the information symbols and further $r_a r_b$ checks on checks. So, the encoding process of a product code can be seen as the serially concatenated application of two components codes. A special feature of this particular case of serial concatenation is that inverting the order of application of the two component codes does not yield any change in the encoded word.

A product code permits to increase the minimum distance in a multiplicative way. If the two component codes have minimum distances d_a and d_b, respectively, the product code has minimum distance $d = d_a \cdot d_b$. Several types of component codes have been used in the design of product codes. SPC codes are often used because of their simplicity, but they can yield severe constraints on the overall code length and rate. Better results can be obtained with product codes based on Hamming codes, that can achieve very good

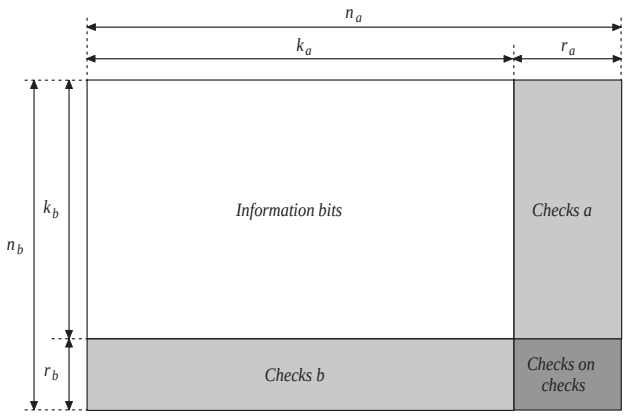

Figure 3. Encoding scheme of a bi-dimensional product code.

performance under soft-input/soft-output iterative decoding [18]. Furthermore, it has been demonstrated that product codes are potentially able to achieve error-free coding with a nonzero code rate (as the number of dimensions increases to infinity) [19], [20].

We are interested in designing bi-dimensional product codes that exploit, as component codes, two serially concatenated codes having the form described in Section II-B. In fact, when large block codes are needed, the form of serial concatenation in Section II-B can require rather large component codes, that can become unpractical for hardware implementation. On the other hand, bi-dimensional product codes could be designed that allow to obtain large blocks while still exploiting component codes with a rather small size. Next, we will show that the parity-check matrix of the bi-dimensional product code can be easily obtained starting from the parity-check matrices of the two component codes, and that the product code is still an LDPC code [1].

Let us suppose that the two component codes have the following parity-check matrices, where $\mathbf{h}_{i,j}$ represents the j-th column of matrix $\mathbf{H}_i$:

$$\begin{aligned} \mathbf{H}_a &= \left[\begin{array}{cccc} \mathbf{h}_{a,1} & \mathbf{h}_{a,2} & \cdots & \mathbf{h}_{a,n_a} \end{array} \right], \\ \mathbf{H}_b &= \left[\begin{array}{cccc} \mathbf{h}_{b,1} & \mathbf{h}_{b,2} & \cdots & \mathbf{h}_{b,n_b} \end{array} \right]. \end{aligned} \tag{9}$$

It follows that $\mathbf{H}_a$ has size $r_a \times n_a$, while $\mathbf{H}_b$ has size $r_b \times n_b$.

A valid parity-check matrix for the product code having such components can be expressed in the following form:

$$\mathbf{H}_p = \left[\begin{array}{c} \mathbf{H}_{p1} \\ \mathbf{H}_{p2} \end{array} \right], \tag{10}$$

where $\mathbf{H}_{p1}$ has size $r_a n_b \times n_a n_b$, and $\mathbf{H}_{p2}$ has size $r_b n_a \times n_a n_b$. $\mathbf{H}_{p1}$ can be obtained as a block-diagonal matrix formed by n_b repetitions of $\mathbf{H}_a$:

$$\mathbf{H}_{p1} = \left[\begin{array}{cccc} \mathbf{H}_a & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_a & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{H}_a \end{array} \right], \tag{11}$$

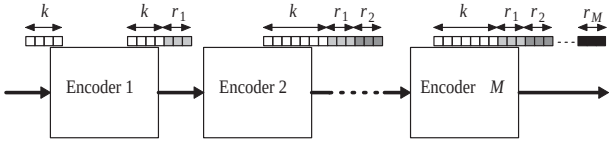

Figure 4. Scheme of the concatenated systematic encoder.

where $\mathbf{0}$ represents an $r_a \times n_a$ null matrix.

$\mathbf{H}_{p2}$ is given by (12): it consists of a row of blocks, in which the i-th block contains, along the main diagonal, n_a copies of $\mathbf{h}_{b,i}$ (that is, the columns of $\mathbf{H}_b$), while all the remaining symbols are null. $\mathbf{H}_p$ is redundant, since it includes two sets of parity-check constraints representing checks on checks through both the component codes. For this reason, $\mathbf{H}_p$ cannot have full rank. When both components are in systematic form, a full rank parity-check matrix for the product code can be obtained by eliminating the last $r_a \cdot r_b$ rows from $\mathbf{H}_{p1}$ or $\mathbf{H}_{p2}$, in such a way to avoid doubled representation of checks on checks.

If we suppose that the densities of 1 symbols in $\mathbf{H}_a$ and $\mathbf{H}_b$ are δ_a and δ_b, respectively, it is easy to prove that the density of $\mathbf{H}_{p1}$ is δ_a / n_b, while that of $\mathbf{H}_{p2}$ is δ_b / n_a. So, even starting from two component codes that are not characterized by very sparse parity-check matrices, the resulting product code can still be an LDPC code. Alternative representations of the parity-check matrix can be found, that can achieve even lower density [21]. For our purposes, however, the density of the parity-check matrix in the form (10), with $\mathbf{H}_{p1}$ and $\mathbf{H}_{p2}$ as expressed by (11) and (12), is low enough.

Furthermore, it is easy to verify that matrix (10) is free of length-4 cycles, provided that the same holds for the component matrices $\mathbf{H}_a$ and $\mathbf{H}_b$. So, the codes obtained as bi-dimensional product codes can be effectively decoded by means of LDPC decoding algorithms. We will give some examples in this sense in the next section.

III. Code Characteristics

A. Encoding and Decoding

The serially concatenated codes we consider, described in Section II-B, can be encoded by using a very simple concatenated encoder structure, like that shown in Figure 4. Each component code, described in Section II-A, is in systematic form; so, the i-th component encoder simply appends r_i redundancy bits to the input vector. The overall codeword results in the concatenation of the input vector and the redundancy vectors added by the cascade of encoders.

Alternatively, the serially concatenated code can be encoded by using the standard "back substitution" technique. For the proposed concatenated code, the low-density parity-check matrix is in lower triangular form; so, the standard encoding algorithm has very low complexity due to the fact that it works on a sparse matrix. For generic LDPC

$$\mathbf{H}_{p2} = \begin{bmatrix} \mathbf{h}_{b,1} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{h}_{b,2} & \mathbf{0} & \cdots & \mathbf{0} & & \mathbf{h}_{b,n_b} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{h}_{b,1} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{h}_{b,2} & \cdots & \mathbf{0} & & \mathbf{0} & \mathbf{h}_{b,n_b} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \cdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{h}_{b,1} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{h}_{b,2} & & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{h}_{b,n_b} \end{bmatrix}. \quad (12)$$

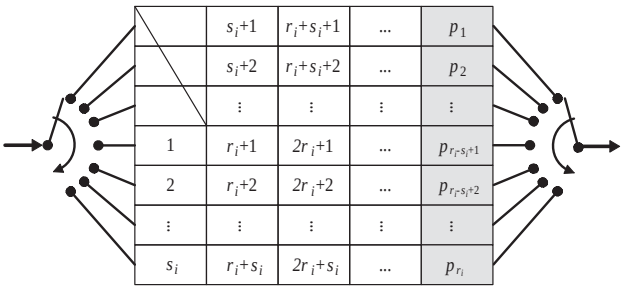

Figure 5. Parallel implementation of the encoder for the i-th component code.

codes, instead, a matrix pre-elaboration through Gaussian elimination may be needed in order to put the parity-check matrix in lower triangular form, and this usually does not preserve its sparse character, thus yielding increased complexity.

On the other hand, when adopting the encoder circuit shown in Figure 4, each component code can be encoded by means of a very simple circuit, based on a linear feedback shift register (LFSR) that implements the polynomial multiplication expressed by Eq. (5). Encoding of each component code can be also implemented by using a parallel encoder architecture and serial to parallel and parallel to serial converters [22]. For the proposed component codes, the parallel encoder coincides with a bank of SPC encoders, as shown in Figure 5. The parallel encoder for the i-th component code can be represented as a binary matrix with r_i rows and $\left\lceil \frac{k_i}{r_i} \right\rceil + 1$ columns. For encoding, its cells are filled in column-wise order, from top left to bottom right. The first $r_i - s_i$ cells, with $s_i = k_i \mathrm{mod} r_i$, are unused, while the others are filled in the order reported in Figure 5, until the first $\left\lceil \frac{k_i}{r_i} \right\rceil$ columns are completed (white cells in the figure). When the j-th row is filled, $j = 1 \ldots r_i$, the parity bit p_j is calculated, by XORing the elements of the row, and its value is stored in the last column, at the same row. When all the parity bits have been calculated, the encoder outputs the codeword by reading the matrix content in the same order used for the input, but including the parity bits.

Due to the LDPC nature of the serially concatenated codes we consider, and to the absence of short cycles in their associated Tanner graph, their decoding can be accomplished through the standard Sum-Product Algorithm with Log-Likelihood Ratios (LLR-SPA) [23], or through its low-complexity versions, like the normalized min-sum (NMS) algorithm [24]. These are BP-based decoding algorithms and, hence, they are able to exploit the length-4 cycle free Tanner graph representation of the code to achieve capacity approaching error-correction performance.

B. Minimum Distance

An upper bound on the minimum distance of the serially concatenated codes can be obtained by exploiting their concatenated nature and the structure of the component codes.

Lemma 2: Serially concatenated codes in the proposed family have minimum distance:

$$d_{\min} \leq 2^M. \quad (13)$$

Proof: Let us consider the concatenated encoder shown in Figure 4 with systematic component encoders as shown in Figure 5, and let us focus on the first code, that has redundancy r_1. Its minimum weight codewords have Hamming weight 2, and correspond to input vectors having weight 1. Due to the encoder structure, the two 1 symbols in each minimum weight codeword are spaced by an integer multiple of r_1, say $a_1 r_1$, with $1 \leq a_1 \leq \left\lfloor \frac{k_1}{r_1} \right\rfloor$, where function $\lfloor \cdot \rfloor$ returns the greatest integer smaller than or equal to its argument.

When such codeword is given as input to the subsequent encoder, the two 1 symbols can be in the same row of the second encoder or not. In the first case, that occurs for $a_1 r_1 = a_2 r_2$, $1 \leq a_2 \leq \left\lfloor \frac{k_2}{r_2} \right\rfloor$, the output codeword has weight 2; otherwise, it has weight 4. The latter case occurs for $k_2 = n_1 < \mathrm{lcm}(r_1, r_2)$, and produces a weight-4 codeword whose 1 symbols can be spaced of integer multiples of r_1, r_2 and linear combinations of them. When such weight-4 codeword is given as input to the third component encoder, its four 1 symbols can be in four different rows or not. In the first case, the Hamming weight is doubled again, thus reaching 8. The same procedure can be generalized by induction, thus obtaining that, for M component codes, the minimum Hamming weight cannot be greater than 2^M, that proves the assertion. ■

The proof of Lemma 2 gives an implicit rule for approaching the upper bound on the minimum distance: the number of coincidences among linear combinations of the r_i values, $i = 1 \ldots M$, must be reduced as much as possible in the range $[1, n]$. The choice of coprime r_i's is also favorable from this viewpoint.

When serially concatenated codes are used as components in bi-dimensional product codes, the minimum distance of the overall code can be easily upper bounded starting from Lemma 2 and considering that the product code structure increases the minimum distance in a multiplicative way. So, the following corollary immediately follows.

Corollary 2: A bi-dimensional product code having, as components, two serially concatenated codes of the considered family, has minimum distance:

$$d_{\min} \leq 2^{M_a+M_b}, \tag{14}$$

where M_a and M_b represent the number of serially concatenated codes in the two components of the product code.

C. Rate Compatibility

Rate compatibility consists in designing a set of "compatible" codes with different rates, in such a way as to allow the implementation of schemes with variable error correction capability.

For the serially concatenated scheme we consider, rate compatibility is ensured by systematic encoding, since redundancy is incrementally appended to the information vector. By considering each component code progressively, a set of rate compatible codes is simply obtained, with code rates

$$\frac{k}{k+r_1} > \frac{k}{k+r_1+r_2} > \ldots > \frac{k}{k+\sum_{j=1}^{M} r_j}. \tag{15}$$

An example of rate compatible serially concatenated codes is reported in the following section. Rate compatibility can also be ensured when the serially concatenated codes we consider are used as components in bi-dimensional product codes. In this case, a simple way to obtain a family of rate compatible codes is to fix one of the two components of the product code, whereas the other one can employ a variable number of its sub-components in order to change the overall code rate.

IV. Examples of Serially Concatenated Codes

In this section, we give some examples of design of serially concatenated codes through the considered technique. We show that such technique is able to produce small, medium and large codes with very fine granularity in the code length and rate, and very good error correction performance. Together with their low complexity encoding, these peculiarities justify possible interest on such codes for practical applications.

All simulations have been performed with Binary Phase Shift Keying (BPSK) modulation over the Additive White Gaussian Noise (AWGN) channel. Coded transmission has been simulated through a suitable software, written in C++ language, that performs a Montecarlo evaluation of bit error rate (BER) and frame error rate (FER) for each value of energy per bit to noise power spectral density ratio (E_b/N_0). In order to provide a sufficient level of confidence for Montecarlo simulations, each BER and FER point has been estimated after waiting the occurrence of 100 erred frames.

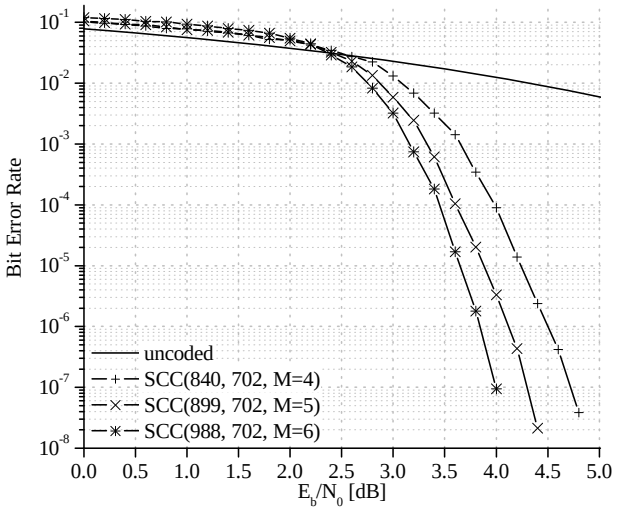

Figure 6. Simulated BER for small codes.

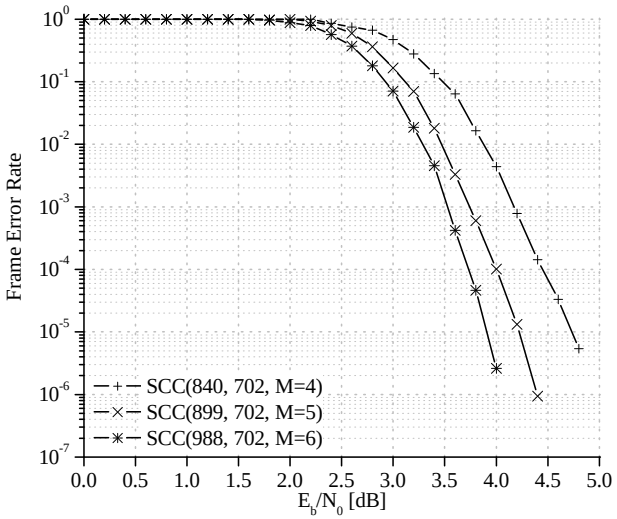

Figure 7. Simulated FER for small codes.

A. Small Size Codes

We consider a first set of rate compatible serially concatenated codes obtained from the following choice of r_i values: $[29, 31, 35, 43, 59, 89]$. All codes in the family have dimension $k = 702$, but different length and rate, depending on the number of component codes. The first code adopts $M = 4$ components, corresponding to the first four values of r_i; thus it has redundancy $r = 138$, length $n = 840$ and rate $R = 0.84$. The second code is obtained by including the fifth component code; so it has redundancy $r = 197$, length

$n = 899$ and rate $R = 0.78$. The last code is obtained by considering all the $M = 6$ values of r_i; thus, it has redundancy $r = 286$, length $n = 988$ and rate $R = 0.71$. Their simulated performance in terms of BER and FER, as a function of the signal-to-noise ratio E_b/N_0, is reported in Figures 6 and 7, respectively.

Rate compatibility of these codes can be exploited in a T-II HARQ scheme, by transmitting, initially, each packet encoded through the first code; then, if requested, by sending r_5 and, finally, r_6 bits of further redundancy.

B. Medium Size Codes

In order to provide a design example of codes with moderate length, we have considered code parameters very similar to those proposed for application in near-Earth space missions by the Consultative Committee for Space Data Systems (CCSDS) [25]. The designed code has length $n = 8208$, dimension $k = 7182$ and, hence, rate $R = 0.875$. Its redundancy ($r = 1026$) corresponds to the following choice of r_i values for the $M = 5$ component codes: $[177, 181, 214, 221, 233]$. Due to the very fine length granularity achievable through the proposed design approach, the code could be arbitrarily shortened in order to have dimension coincident with an integer multiple of 32, as suggested in [25].

The error correction performance of the proposed code, reported in Figure 8, looks very good: its curves are almost overlaid with those of an optimized code with very similar parameters (it has length $n = 8176$ and dimension $k = 7156$) recommended by the CCSDS. The performance of the latter code in terms of BER and FER, shown in Figure 8 (and derived from [25]), refers to an FPGA implementation adopting a maximum number of decoding iterations equal to 50.

It should be observed that the proposed code, besides achieving almost the same performance as the CCSDS code, allows the implementation of very simple encoder circuits, due to its concatenated nature. Therefore, it provides a valid alternative to the CCSDS code that, in turn, is characterized by low encoding complexity thanks to its quasi-cyclic nature.

C. Large Size Codes

As a further example, we have considered the design of large codes with high rate. We have adopted the following choice of r_i values for the $M = 5$ component codes: $[313, 569, 577, 641, 643]$, that yields redundancy $r = 2743$. With this choice of the parameters, we have designed a first code with length $n = 27430$, i.e. dimension $k = 24687$ and rate $R = 0.9$. Subsequently, this code has been shortened to length $n = 10972$, thus producing a code with dimension $k = 8299$ and rate $R = 0.75$.

In order to assess the effect of a different value of M, we have also considered an alternative code design that gives the same value of redundancy ($r = 2743$), but exploiting

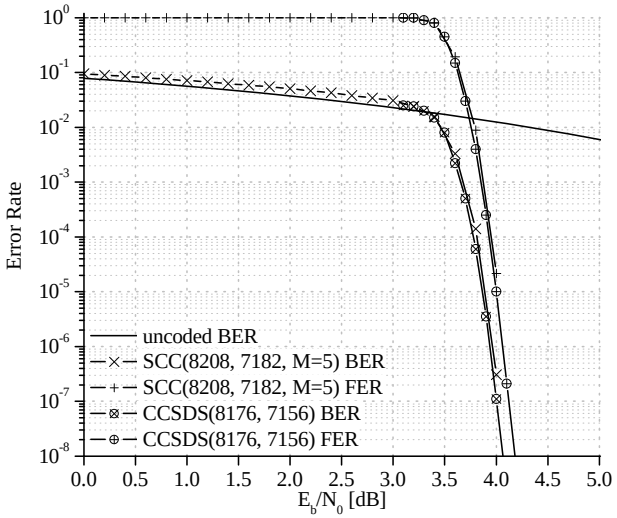

Figure 8. Performance of medium codes.

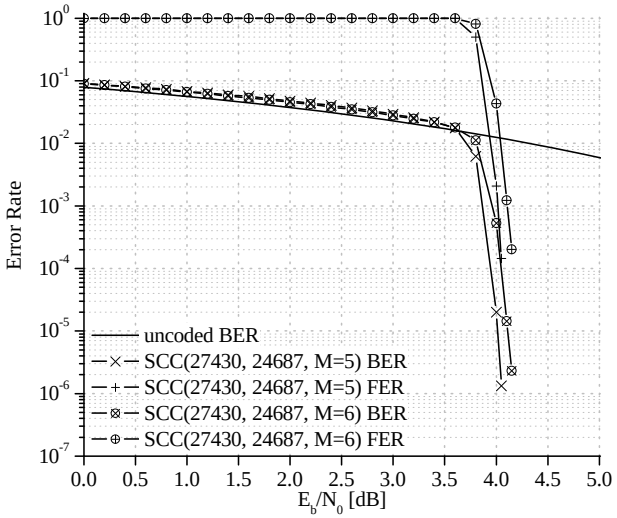

Figure 9. Performance of large codes with rate 0.9.

$M = 6$ serially concatenated codes. In this case, the chosen r_i values are: $[313, 349, 401, 479, 563, 638]$.

Figure 9 shows the simulated performance of the two codes with $n = 27430$ and rate 0.9. As we notice from the figure, the adoption of a higher number of component codes, in this case, does not give any advantage. In particular, both codes do not show any error floor in the explored region. The serially concatenated code with $M = 5$ components, however, has better performance in the waterfall region, and its BER curve intersects that of an uncoded transmission at a lower signal-to-noise ratio. So, it could be concluded that there is no need to increase the number of components over 5 for large codes with such a high rate.

However, when the code is shortened in such a way as to achieve lower rate, such conclusion is no more valid.

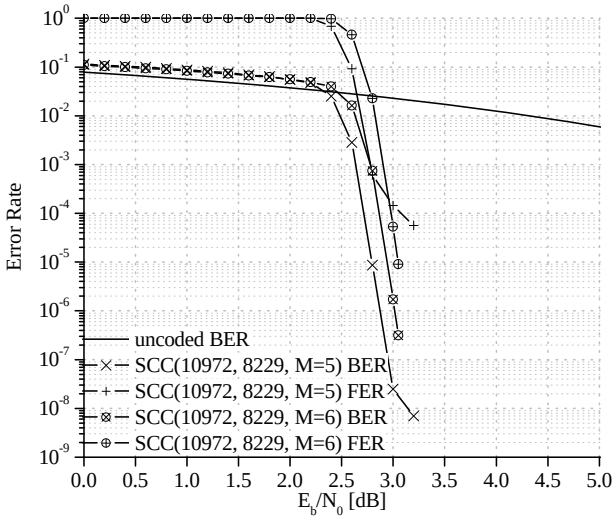

Figure 10. Performance of large codes with rate 0.75.

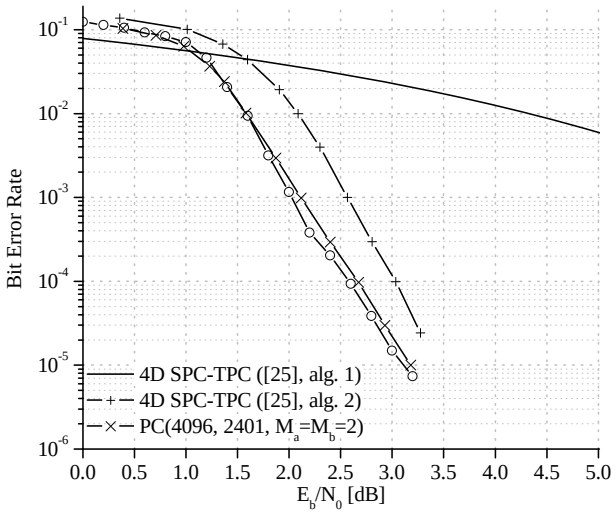

Figure 11. Comparison between $(4096, 2401)$ SPC-TPC and a product code with the same parameters.

The simulated performance of the two codes with length $n = 10972$ and rate 0.75 is reported in Figure 10. As we notice from the figure, an error floor appears in the curves of the code with $M = 5$ components, though at $\mathrm{BER} < 10^{-7}$; on the other hand, the waterfall behavior of the code is very good. The presence of an error floor can be avoided, at the cost of a worse waterfall performance, by increasing the number of serially concatenated components up to $M = 6$. In this case, the error rate curves do not exhibit any change in their slope; so, they can intersect those of the code with $M = 5$ components. The FER curve for $M = 6$ intersects that for $M = 5$ at $\mathrm{FER} \simeq 10^{-4}$; an intersection is also expected for the BER curve.

V. Examples of Product Codes

In this section, we provide some examples of bi-dimensional product codes that exploit, as components, two serially concatenated codes of the type described in Section II-B. For conciseness, we denote them as "product M-SC-MPC codes". The first example we consider is a product code with equal components. More precisely, each component code is a serially concatenated code having length $n_a = n_b = 64$, dimension $k_a = k_b = 49$ and $r_1 = 7$, $r_2 = 8$. Hence, the product code has length $n = 4096$, dimension 2401 and rate 0.586. Figure 11 shows its simulated performance by using the log-likelihood version of the SPA decoding algorithm. For the sake of comparison, the figure shows the performance of a 4D-TPC based on $(8, 7)$ SPC components [26], that has exactly the same parameters.

We observe that the bi-dimensional product code outperforms the 4D-SPC-TPC, when the latter is decoded through algorithm 1 in [26]. Instead, when adopting the decoding algorithm 2 in the same reference, performance of the two codes is almost the same. However, an important difference between the two codes is the fact that our product code is bi-dimensional, while the SPC-TPC is quadri-dimensional and, therefore, has larger decoding latency and complexity.

Two further examples are given in Figures 12 (for the BER) and 13 (for the FER), where two larger product M-SC-MPC codes are considered. The first code has $(n, k) = (10000, 5670)$ and has been obtained as the product of two different serially concatenated codes. Their parameters are as follows: $n_a = 100$, $k_a = 81$ and $r_j^a = [9, 10]$; $n_b = 100$, $k_b = 70$ and $r_j^b = [7, 11, 12]$.

The second code has $(n, k) = (12544, 6400)$; so, its rate is slightly reduced with respect to the former one. It adopts, as components, twice the same M-SC-MPC code. The latter has length $n_a = n_b = 112$, dimension $k_a = k_b = 80$ and $r_j^a = r_j^b = [8, 11, 13]$.

From the simulation results we see that, as expected, by using larger component codes, product M-SC-MPC codes can achieve better performance. The higher number of serially concatenated codes in each component allows to increase the minimum distance and improve performance in the error floor region. This is evident for the $(12544, 6400)$ code, whose components are both based on an order-3 serial concatenation. Together with its slightly lower rate, this makes the performance of such code better than that of the $(10000, 5670)$ code. It should also be noted that, differently from SPC-TPCs, in our codes the increase in minimum distance is achieved though preserving the bi-dimensional nature of the product code.

In order to better assess the effect on performance of the choice of M_a and M_b, that is, the number of component codes in each one of the two serially concatenated codes forming the product code, we have designed three further

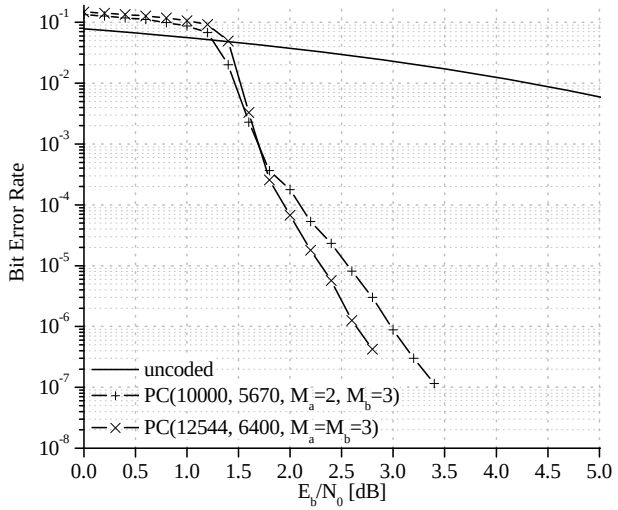

Figure 12. BER performance for $(10000, 5670)$ and $(12544, 6400)$ product codes.

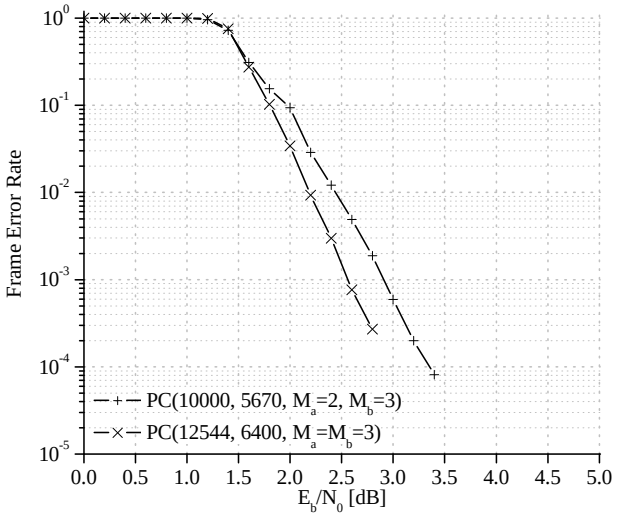

Figure 13. FER performance for $(10000, 5670)$ and $(12544, 6400)$ product codes.

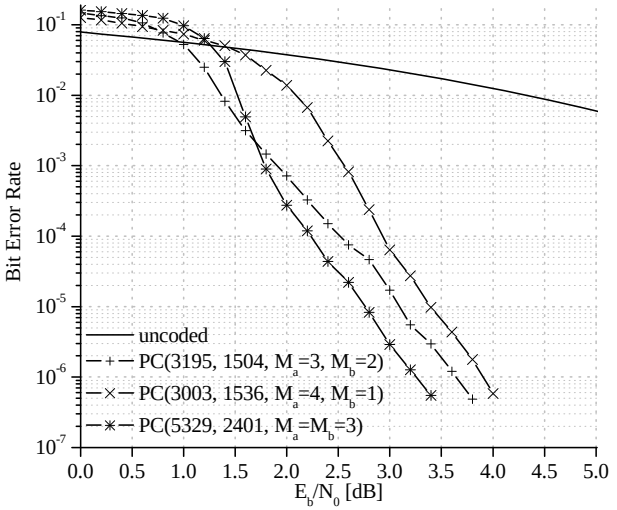

Figure 14. BER performance for $(3195, 1504)$, $(3003, 1467)$ and $(5329, 2401)$ product codes.

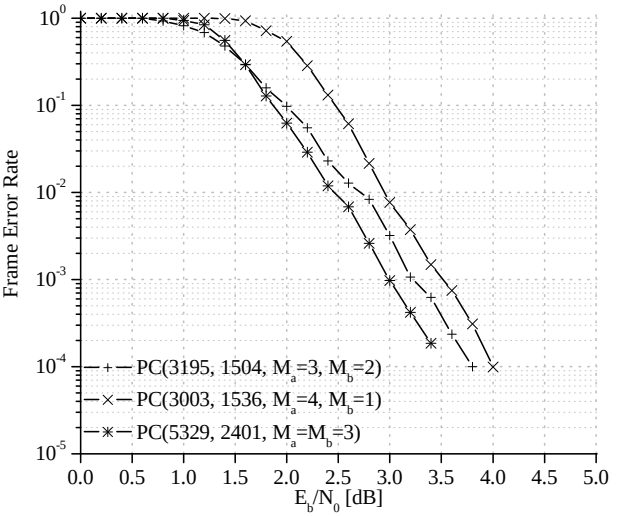

Figure 15. FER performance for $(3195, 1504)$, $(3003, 1467)$ and $(5329, 2401)$ product codes.

product codes with rate around $1/2$ and comparable size. The first code has length 3195 and dimension 1504; its two components have $n_a = 71$, $k_a = 47$, $r_j^a = [7, 8, 9]$ and $n_b = 45$, $k_b = 32$, $r_j^b = [6, 7]$, respectively. So, in this case, we have fixed $M_a = 3$ and $M_b = 2$. In the design of a second product code, we have adopted a different distribution of the serially concatenated components, that is, $M_a = 4$ and $M_b = 1$. The product code has length $n = 3003$ and dimension $k = 1467$. The first one of its two component codes has $n_a = 91$, $k_a = 48$ and $r_j^a = [9, 10, 11, 13]$. The second component, instead, is a single parity-check code with $n_b = 33$ and $k_b = 32$. As a further example, we have designed a third product code with $M_a = M_b = 3$. It uses twice the same serially concatenated code, that has length $n_a = n_b = 73$, dimension $k_a = k_b = 49$ and $r_j^a = r_j^b = [7, 8, 9]$. Thus, the product code has length $n = 5329$ and dimension $k = 2401$.

The simulated performance is reported in Figure 14, for the BER, and in Figure 15, for the FER. It results from numerical simulations that the code with $M_a = 3$ and $M_b = 2$ has better performance with respect to the code having $M_a = 4$ and $M_b = 1$, especially in the waterfall region. This suggests that a balanced choice of M_a and M_b can yield a performance improvement with respect to an unbalanced choice.

If both M_a and M_b are increased up to 3, performance can be further improved, at the cost of a larger code block, due to the constraints of the product structure. However, the

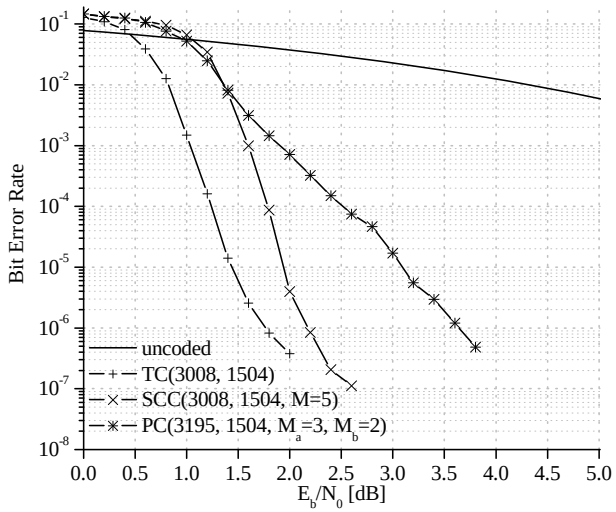

Figure 16. BER performance for codes with DVB-RCS compliant parameters.

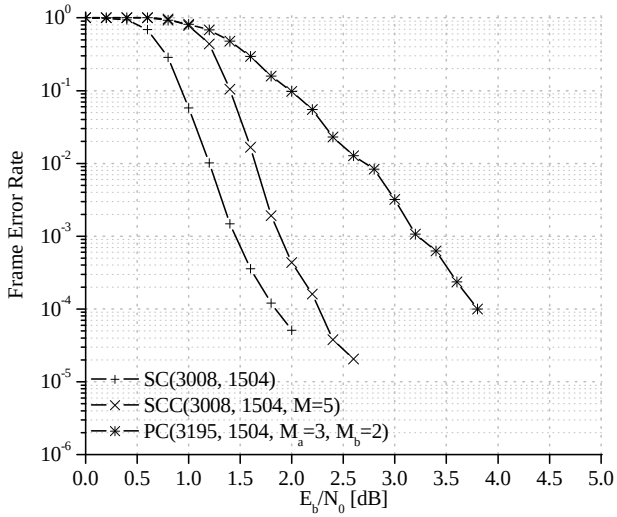

Figure 17. FER performance for codes with DVB-RCS compliant parameters.

$(5329, 2401)$ code is based on two very small component codes, so its complexity could still be reasonably low for many practical applications.

In comparison with single serially concatenated codes, it is reasonable to expect that the adoption of smaller component codes in a product code is paid in terms of a worse error rate performance. In order to verify such conclusion, we have considered a set of codes having parameters compliant with the DVB-RCS standard [27]. We have focused on the option of MPEG 2 frames, that are 188 bytes long, and code rate $R = 1/2$. For this choice of the parameters, the standard recommends the usage of a $(3008, 1504)$ double binary turbo code with an optimized interleaver.

We have designed a serially concatenated code with exactly the same parameters. It has length 3008 and adopts the following choice of the r_i's for its components: $[277, 281, 293, 313, 340]$. For the sake of comparison, we have also considered one of the product codes introduced in the previous section. It has dimension 1504, like in the standard, but length 3195, because of the constraints due to the product code design. Its components have $n_a = 71$, $k_a = 47$ and $r_j^a = [7, 8, 9]$; $n_b = 45$, $k_b = 32$ and $r_j^b = [6, 7]$.

The simulated performance of the serially concatenated and the product code is reported in Figure 16, for the BER, and in Figure 17, for the FER. The simulated performance of the standard turbo code is also reported as a benchmark.

From the figures we see that the product code requires higher SNR per bit than the single serially concatenated code with $M = 5$; the loss at $\mathrm{BER} \simeq 10^{-6}$ and $\mathrm{FER} \simeq 10^{-4}$ is in the order of 1.5 dB. This is the price to pay for reducing complexity by adopting a product code structure. In this example, where we have focused on low rate codes, the standard turbo code achieves the best performance, also in comparison with the M-SC-MPC code. On the contrary, at higher code rates, M-SC-MPC codes and, more generally, structured LDPC codes can be able to outperform turbo codes [28].

VI. Conclusion

We have shown that the concatenation of very simple component codes can be effectively exploited in the design of structured LDPC codes. This allows to obtain a family of LDPC codes characterized by fine length and rate granularity, low complexity and rate compatibility. They ensure good error correction performance, and compete with codes optimized for specific applications. When large block codes are needed, a solution for continuing to exploit small components is to use serially concatenated codes in bi-dimensional product code structures. We have considered bi-dimensional product codes obtained as the direct product of two serially concatenated codes. We have shown that such product codes are still LDPC codes, and that their associated Tanner graph is suitable for performing LDPC decoding. Our simulations show that LDPC product codes in the considered class are able to achieve rather good performance in spite of their very low complexity. However, the advantage of using very small components in product code structures is paid in terms of coding gain: simulation results show a loss on the order of 1.5 dB for LDPC product codes, compared with single LDPC codes with the same parameters.

As a further work, the introduction of an interleaver within the product code structure could be considered, in order to optimize the decoder performance in the waterfall region while preserving the multiplicative effect on the minimum distance due to the product code structure.

REFERENCES

[1] M. Baldi, G. Cancellieri, and F. Chiaraluce, "A class of low-density parity-check product codes," in *Proc. SPACOMM 2009*, Colmar, France, Jul. 2009, pp. 107–112.

[2] G. D. Forney, Jr., "Concatenated codes," Ph.D. dissertation, MIT Press, Cambridge, MA, 1966.

[3] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes," in *Proc. IEEE International Conference on Communications (ICC '93)*, vol. 2, Geneva, Switzerland, May 1993, pp. 1064–1070.

[4] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. 20, no. 2, pp. 284–287, Mar. 1974.

[5] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inform. Theory*, vol. IT-8, pp. 21–28, Jan. 1962.

[6] C. Sae-Young, G. Forney, T. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit," *IEEE Commun. Lett.*, vol. 5, no. 2, pp. 58–60, Feb. 2001.

[7] D. J. C. MacKay and R. M. Neal, "Good codes based on very sparse matrices," in *Cryptography and Coding. 5th IMA Conference*, ser. Lecture Notes in Computer Science, C. Boyd, Ed. Berlin: Springer, 1995, no. 1025, pp. 100–111.

[8] M. P. C. Fossorier, "Quasi-cyclic low-density parity-check codes from circulant permutation matrices," *IEEE Trans. Inform. Theory*, vol. 50, no. 8, pp. 1788–1793, Aug. 2004.

[9] X. Y. Hu, E. Eleftheriou, and D. M. Arnold, "Regular and irregular progressive edge-growth Tanner graphs," *IEEE Trans. Inform. Theory*, vol. 51, no. 1, pp. 386–398, Jan. 2005.

[10] M. Baldi, G. Cancellieri, and F. Chiaraluce, "New LDPC codes based on serial concatenation," in *Proc. AICT '09*, Venice/Mestre, Italy, May 2009, pp. 31–315.

[11] ——. (2009) Good LDPC codes based on very simple component codes. [Online]. Available: http://www.gtti.it/GTTI09/Programma.php

[12] J. Hagenauer, "Rate-compatible punctured convolutional codes (RCPC codes) and their applications," *IEEE Trans. Commun.*, vol. 36, no. 4, pp. 389–400, Apr. 1988.

[13] F. Babich, G. Montorsi, and F. Vatta, "Rate-compatible punctured serial concatenated convolutional codes," in *Proc. IEEE Globecom'03*, vol. 4, San Francisco, CA, Dec. 2003, pp. 2062–2066.

[14] ——, "Performance enhancement of partially systematic rate-compatible SCCCs through puncturing design," in *Proc. IEEE Globecom'04*, vol. 1, Dallas, TX, Nov. 2004, pp. 167–171.

[15] S. B. Wicker, *Error Control Systems for Digital Communication and Storage*. Prentice Hall, 1994.

[16] M. Baldi, G. Cancellieri, A. Carassai, and F. Chiaraluce, "LDPC codes based on serially concatenated multiple parity-check codes," *IEEE Commun. Lett.*, vol. 13, no. 2, pp. 142–144, Feb. 2009.

[17] J. S. K. Tee, D. P. Taylor, and P. A. Martin, "Multiple serial and parallel concatenated single parity-check codes," *IEEE Trans. Commun.*, vol. 51, no. 10, pp. 1666–1675, Oct. 2003.

[18] F. Chiaraluce and R. Garello, "Extended Hamming product codes analytical performance evaluation for low error rate applications," *IEEE Trans. Wireless Commun.*, vol. 3, no. 6, pp. 2353–2361, Nov. 2004.

[19] P. Elias, "Error free coding," *IRE Trans. Inf. Theory*, vol. 4, no. 4, pp. 29–37, Sep. 1954.

[20] D. M. Rankin, T. A. Gulliver, and D. P. Taylor, "Asymptotic performance of single parity-check product codes," *IEEE Trans. Inform. Theory*, vol. 49, no. 9, pp. 2230–2235, Sep. 2003.

[21] M. Esmaeili, "Construction of binary minimal product parity-check matrices," *Applicable Algebra in Engineering, Communication and Computing*, vol. 19, no. 4, pp. 339–348, Aug. 2008.

[22] O. Gazi and A. O. Yilmaz, "On parallelized serially concatenated codes," in *Proc. WCNC 2007*, Hong Kong, Mar. 2007, pp. 714–718.

[23] J. Hagenauer, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," *IEEE Trans. Inform. Theory*, vol. 42, no. 2, pp. 429–445, Mar. 1996.

[24] J. Chen and M. Fossorier, "Near optimum universal belief propagation based decoding of low-density parity check codes," *IEEE Trans. Commun.*, vol. 50, pp. 406–414, Mar. 2002.

[25] CCSDS, "Low Density Parity Check Codes for use in Near-Earth and Deep Space Applications," Consultative Committee for Space Data Systems (CCSDS), Washington, DC, USA, Tech. Rep. Orange Book, Issue 2, Sep. 2007, CCSDS 131.1-O-2.

[26] D. M. Rankin and T. A. Gulliver, "Single parity check product codes," *IEEE Trans. Commun.*, vol. 49, no. 8, pp. 1354–1362, Aug. 2001.

[27] *Digital Video Broadcasting (DVB); Interaction channel for satellite distribution systems*, ETSI EN Std. 301 790 (v1.4.1), Sep. 2005.

[28] M. Baldi, F. Chiaraluce, and G. Cancellieri, "Finite-precision analysis of demappers and decoders for LDPC-coded M-QAM systems," *IEEE Trans. Broadcast.*, vol. 55, no. 2, pp. 239–250, Jun. 2009.